32192/32195/32196 Group

Application of the TOU (3 Phase motor control)

1. Overview

This documentation presents examples of sample program, which use the Timer TOU incorporated in the 32192/32195/32196 Group Microcomputer.

2. Introduction

The sample task described in this documentation applies to the use of the following microcomputers, under the respective conditions.

- Microcomputer: 32192 Group (M32192F8VFP, M32192F8UFP, and M32192F8TFP)
  32195 Group (M32195F4VFP, M32195F4UFP, and M32195F4TFP)
  32196 Group (M32196F8VFP, M32196F8UFP, and M32196F8TFP)
- Operating Frequency: 128 to 160 MHz (The sample programs are compiled assuming a frequency of 160 MHz.)
- Operation Board: Starter kit for 32192

3. Explanation of the Technology Applied

3.1 Outline of the Multijunction Timer

The Multijunction Timer (MJT following MJT for abbreviation) have Input event bus and Output event bus, it can be single used and internal connection is possible mutually.

This function allows constructing Timer flexibly and corresponding to various applications. Multijunction is named because timers have many connection points with internal event bus.

Please refer to the hardware manual for details of the Multijunction Timer.

4. Phase-compensated PWM waveform output Sample Program

4.1 Outline of the Sample Program

In this sample program, TOU0_0 to TOU0_5 (PWM waveform output), TOU0_6 (calculates for next cycle and set that value), TOU0_7 (for generating carrier cycle) and outputted phase-compensated PWM waveform for 3 phase. The waveform is outputted to external pins TO21 to TO26 (using pin group A). In addition, PWM Output prohibits function that cuts output during detecting simultaneous "L" level at positive and negative phase of phase-compensated PWM waveform is used.

In this sample program Sine table divided by 256 is used, increment every 20kHz and triangle of 78.125 Hz outputs PWM waveform by control.

Figure 4.1.1 shows phase-compensated PWM waveform for 3 phase used in 3 phase motor control.

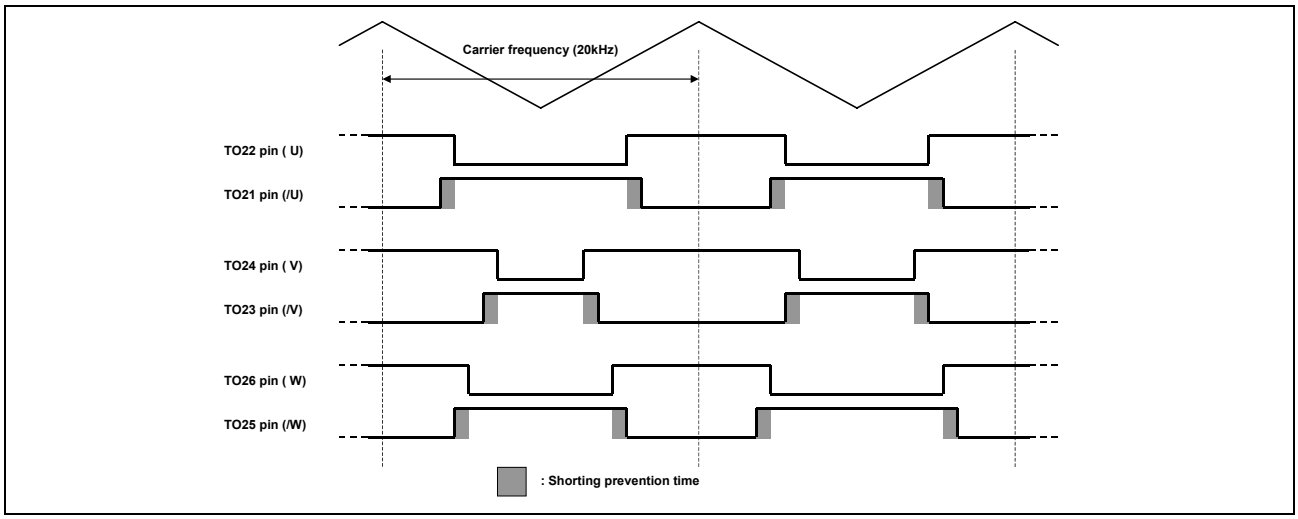


Figure 4.1.1 Output waveform

4.2 Timer construction

Because count source of timer TOU is maximum of PWM resolution, select BCLK (40MHz) and use with prescaler3 divided by 1. TOU0_7 set to continuous output mode, generate 20kHz of carrier cycle. TOU0_0, TOU0_2, TOU0_4 are set to single-shot PWM output mode, TOU0_6 is set to single-shot output mode and TOU0_0, TOU0_2, TOU0_4, TOU0_6 are started by the underflow of TOU0_7.

TOU0_1, TOU0_3, TOU0_5 are set to single-shot output mode, use for shorting prevention function is used.

PWM output waveform generated in TOU0_0 to TOU0_5 is outputted from TO21 to TO26.

Set the value underflow in the point 3/4 of carrier cycle for TOU0_6.

In addition, use input from external pin and PWM output disable function by port pin level.

Figure 4.2.1 shows Timer construction, and Figure 4.2.2 shows Operation timing.

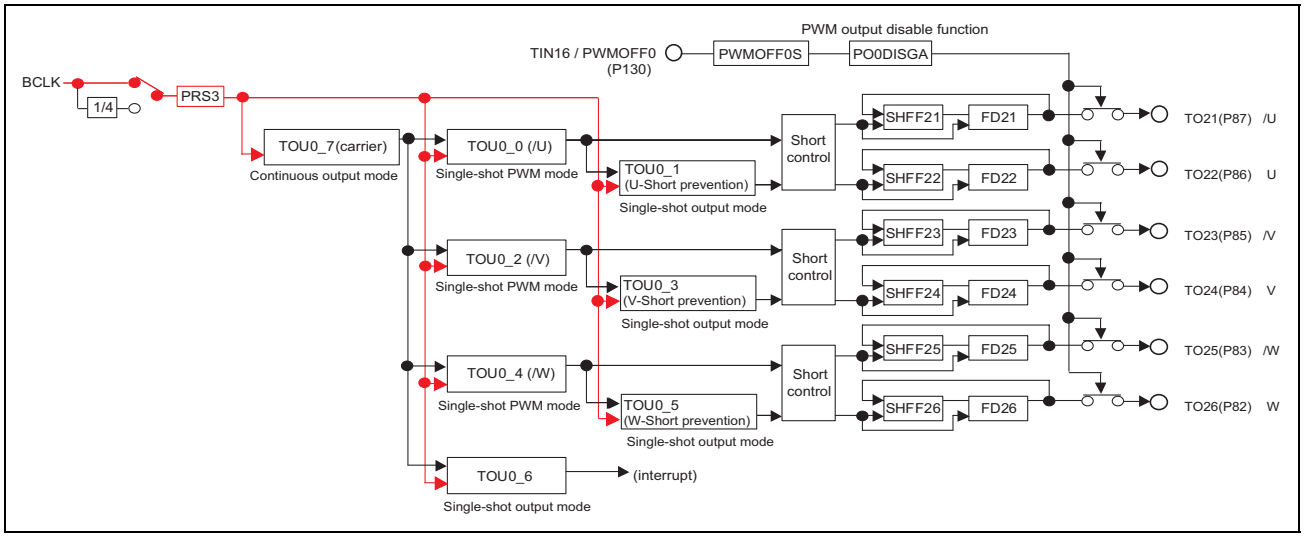


Figure 4.2.1 Timer construction

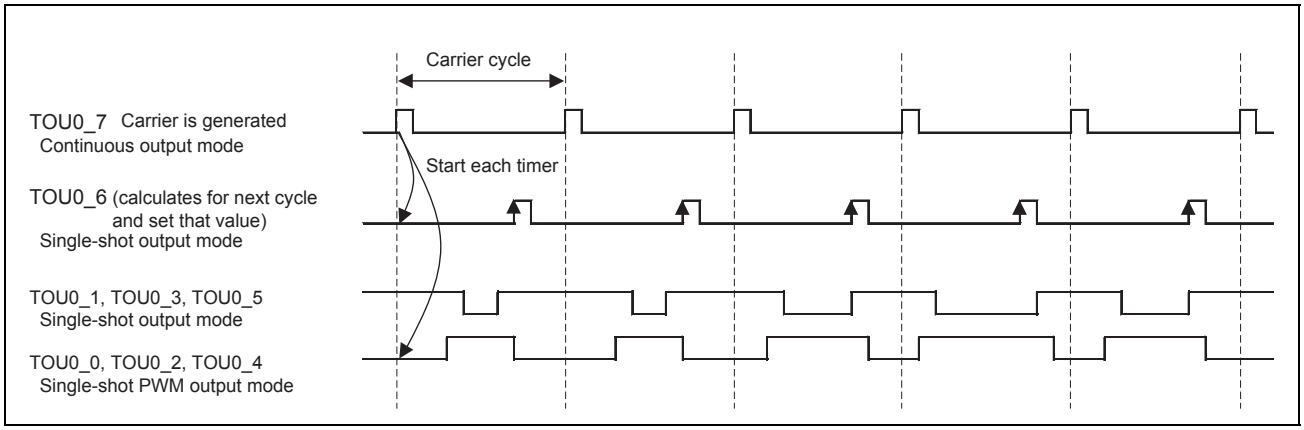


Figure 4.2.2 Operation timing

4.3 Timer operation

4.3.1 Calculation way of output waveform duty

Duty calculation of PWM waveform and setting register of TOU0_0, TOU0_2 and TOU0_4 is operated by TOU0_6 interrupt processing.
Shorting prevention time operates setting reload register of TOU0_1, TOU0_3 and TOU0_5.

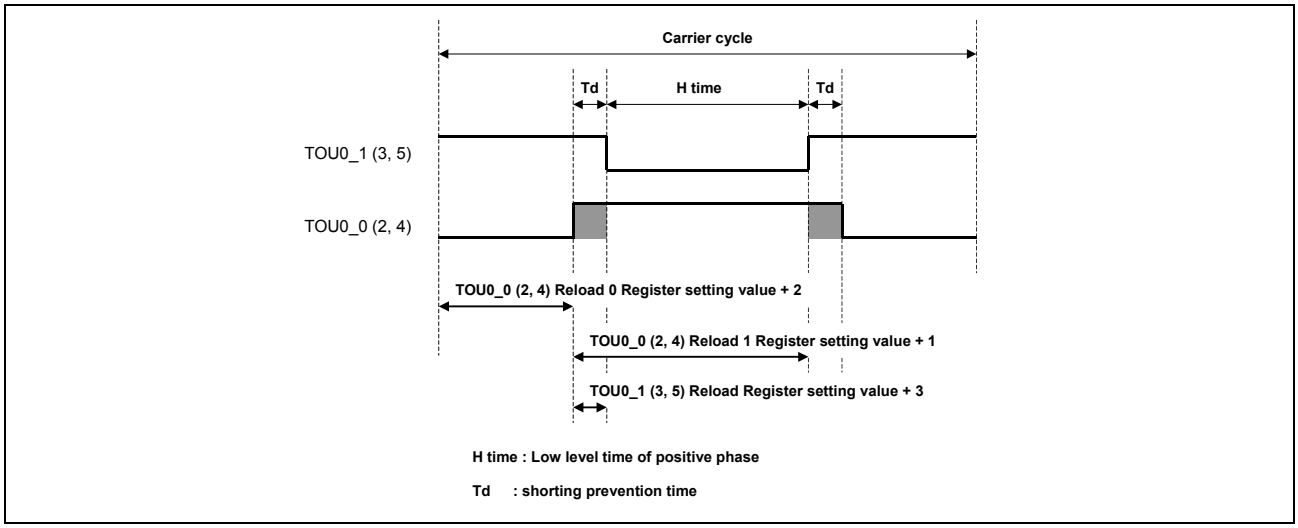

Carrier cycle
Td
H time
Td
TOU0_1 (3, 5)
TOU0_0 (2, 4)
TOU0_0 (2, 4) Reload 0 Register setting value + 2
TOU0_0 (2, 4) Reload 1 Register setting value + 1
TOU0_1 (3, 5) Reload Register setting value + 3
H time : Low level time of positive phase
Td : shorting prevention time

Figure 4.3.1 Calculation way of output waveform duty

When outputs waveform like figure 4.3.1, reload register setting value of each timer can be calculated by following equations.

TOU0_1 (3, 5):
Reload register = Td -3
TOU0_0 (2, 4):
Reload 1 register = Htime + Td -1
Reload 0 register = ((Carrier cycle - Htime) /2) - Td -2

4.3.2 Operation of short prevention function

When short prevention function is valid, TOU0 enable factor select bit of TOU0_1 (3, 5) is invalid, those are started by underflow of TOU0_0 (2, 4).

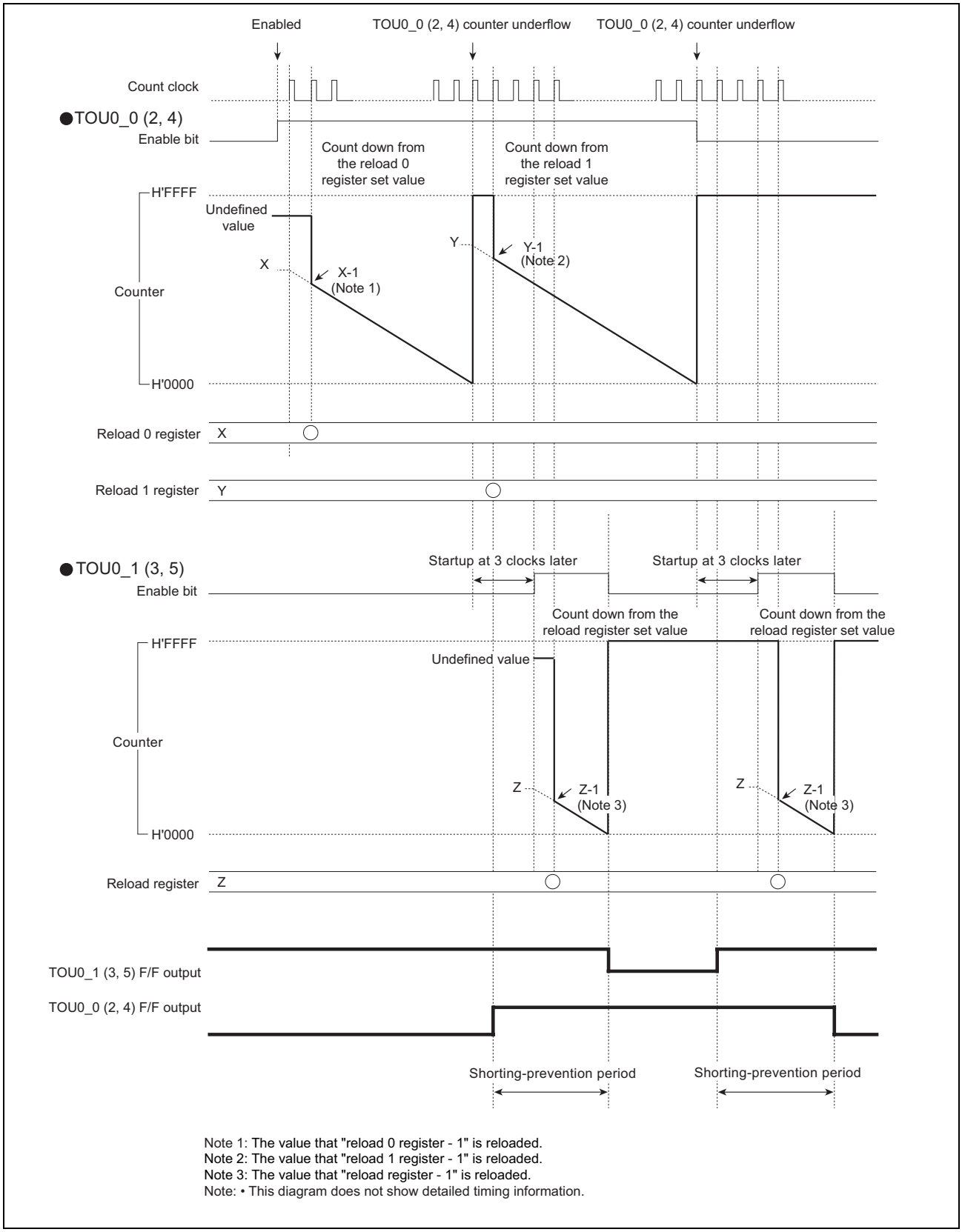
TOU0_0 (2, 4) is started by underflow of TOU0_7 and “reload 0 register setting value -1”is loaded to counter with synchrony count clock, start down counting. “Reload 1 register setting value -1” is loaded and start down counting at first underflow. Then stop count at second underflow. After 3 count clock of first and second counter underflow, TOU0_1 (3, 5) is started.

Figure 4.3.2 shows Timer operation when short prevention function is valid and diagram of output waveform.

When short prevention function is valid, it is necessary to set value in F/F data register and F/F data register for short prevention function.
When reversing of F/F output, the value of the F/F data register and the F/F data register for short prevention function is replaced, and the value of the F/F data register after the change becomes the output of F/F.
Furthermore, if sets same value to F/F data register and F/F data register for short prevention function, it is going to be fixed output.

- High level is outputted at first
  Set F/F data register to “1”, F/F data register for short prevention function to “0”.
- Low level is outputted at first
  Set F/F data register to “0”, F/F data register for short prevention function to “1”.
- Output of fixed
  The same value to the F/F data register and the F/F data register for short prevention function is set.
  When "0" is set to each register, "L" level is outputted, "1" it sets when, "H" level is outputted.

Note. When short prevention function invalid/valid bit is set, counter of all TOU timers should be count stopped.



Note 1: The value that "reload 0 register - 1" is reloaded.
Note 2: The value that "reload 1 register - 1" is reloaded.
Note 3: The value that "reload register - 1" is reloaded.
Note: • This diagram does not show detailed timing information.

Figure 4.3.2 Timer operation when short prevention function is valid

4.3.3 Setting F/F data register, F/F data register for short prevention function and pin output

Setting value of F/F data register, F/F data register for short prevention function and pin output are shown below.

In this sample program, the setting that output "H" to positive phase side, "L" to negative phase side is used.

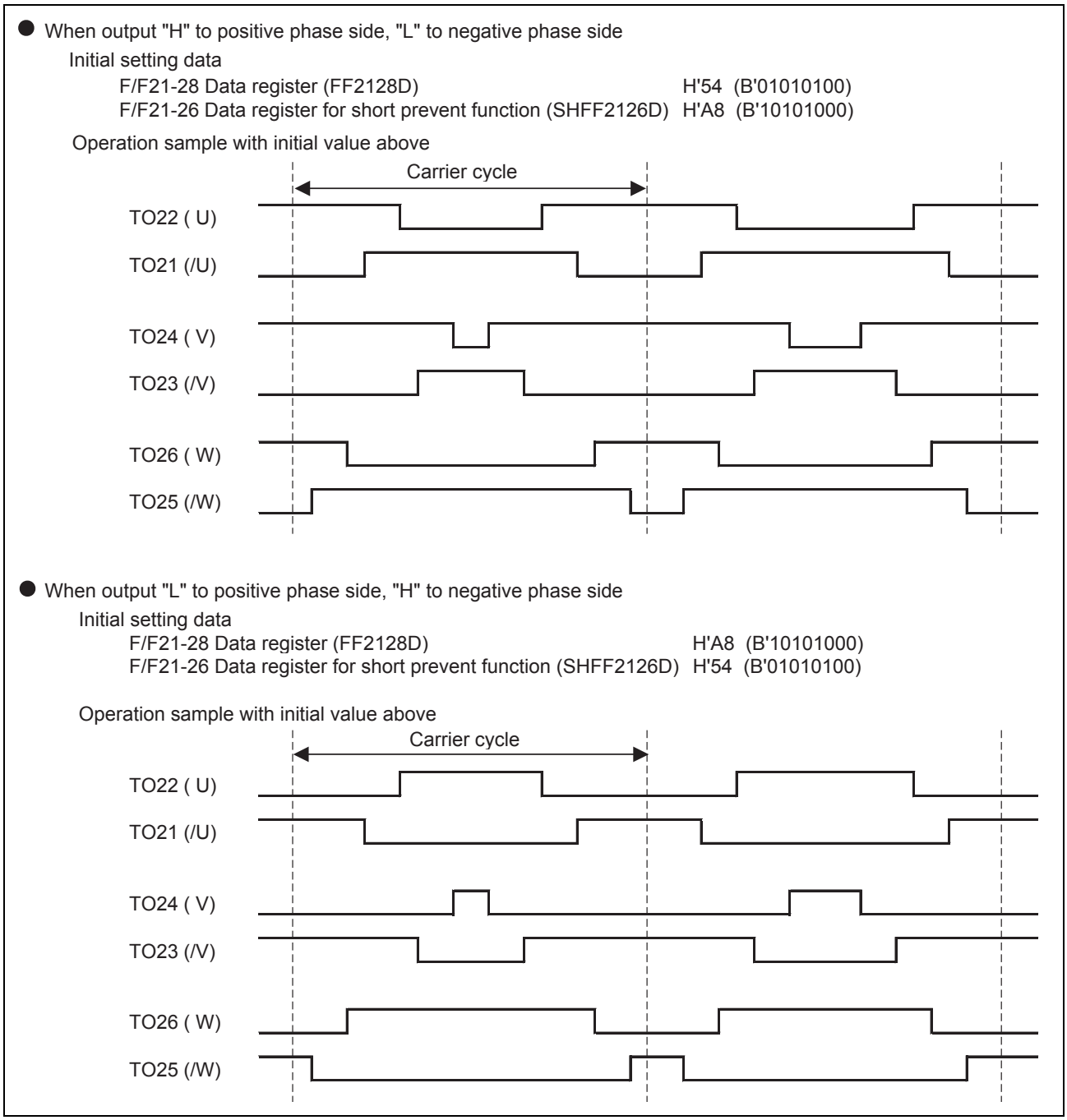

When output "H" to positive phase side, "L" to negative phase side
Initial setting data
F/F21-28 Data register (FF2128D) H'54 (B'01010100)
F/F21-26 Data register for short prevent function (SHFF2126D) H'A8 (B'10101000)
Operation sample with initial value above
Carrier cycle
TO22 (U)
TO21 (/U)
TO24 (V)
TO23 (/V)
TO26 (W)
TO25 (/W)
When output "L" to positive phase side, "H" to negative phase side
Initial setting data
F/F21-28 Data register (FF2128D) H'A8 (B'10101000)
F/F21-26 Data register for short prevent function (SHFF2126D) H'54 (B'01010100)
Operation sample with initial value above
Carrier cycle
TO22 (U)
TO21 (/U)
TO24 (V)
TO23 (/V)
TO26 (W)
TO25 (/W)

Figure 4.3.3 Relation between F/F data registers and pin output

4.3.4 Setting Duty 0%, Duty 100%

With this sample program, Duty 0%, or as for Duty 100% output, data setting timing of the next period (TOU0_6 interrupting processing) the time, the F/F data register for short prevention function (SHFF2126D) it actualizes by the fact that value is operated.
As the F/F data register by the fact that the same value is set, "L" level, or "H" level is outputted in the F/F data register for short prevention function.

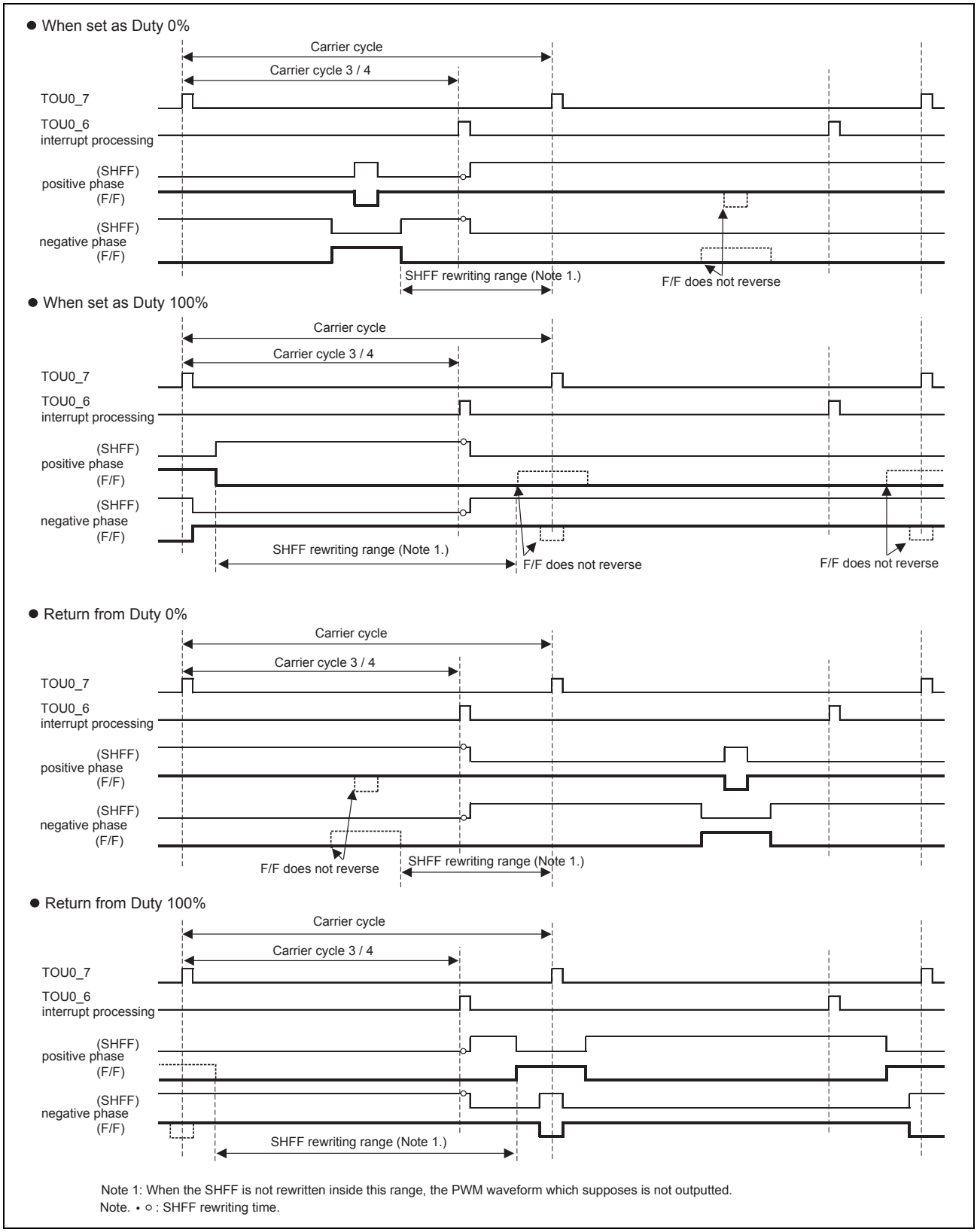
To secure the SHFF rewriting time, at carrier period at the time of Duty 0% setting, the PWM waveform that is close to 0% is outputted, at carrier period at the time of Duty 100% setting, the PWM waveform that is close to 100% is outputted.

In Duty 0% the value which outputs the PWM waveform of the smallest width, the inside of Duty 100% has set the value which outputs the PWM waveform of breadth maximum to each reload registers.

With this sample program, because it is necessary to do in "SHFF rewriting range" in figure 4.3.4, as generated the interruption of the TOU0_6 at 3/4 positions of carrier period, it sets the modification of F/F data register for short prevention function value.

TOU0_6 interruption occurrence timing has the necessity to consider the fact that the delay other interruption processing and the like with influence occurs.

Figure 4.3.4 shows the timing diagram of setting Duty 0%, Duty 100%.



Note 1: When the SHFF is not rewritten inside this range, the PWM waveform which supposes is not outputted.
Note. • ○ : SHFF rewriting time.

Figure 4.3.4 The timing diagram of setting Duty 0%, Duty 100%

4.3.5 PWM output prohibit function

32192/32195/32196 group has function which can prohibit output from P87 (P00)/TO21 to P82 (P05)/TO26 and P110(P10)/TO29 to P115(P15)/TO34 (those are external pin of TOU0_0 to TOU0_5 timer and TOU1_0 to TOU1_5 timer) compulsorily.

To prohibit outputting is following three ways.

- PWM output is prohibited by signal inputted from external pin (TIN16/PWMOFF0, TIN17/PWMOFF1)
- PWM output is prohibited by PWM output prohibit control register
- PWM output is prohibited by pin level of port P87 (P00)/ TO21 to P82 (P05)/ TO26, P110 (P10)/ TO29 to P115 (P15)/ TO34

Figure 4.3.5 shows circuit construction diagram of PWM output prohibit function.

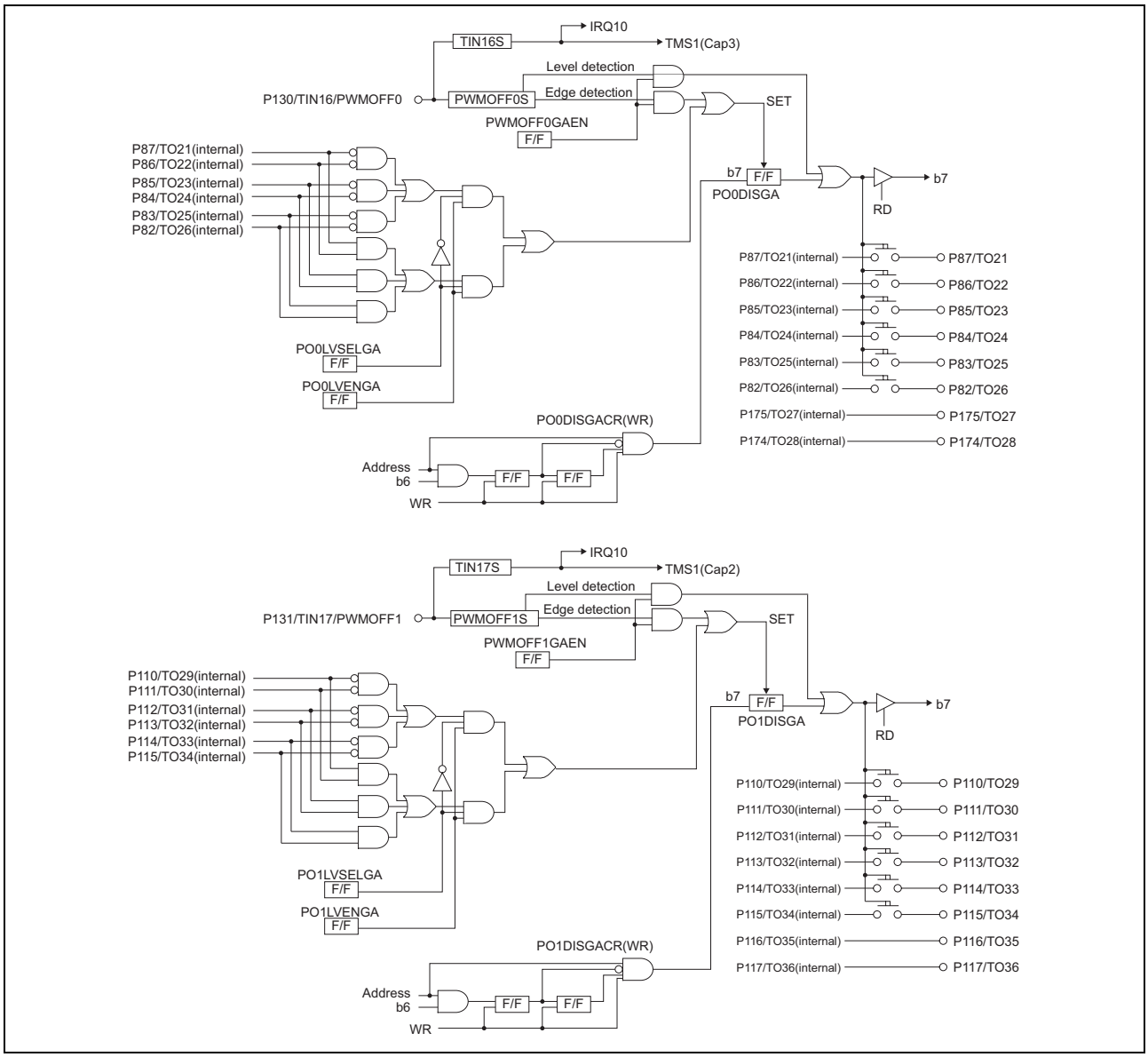


Figure 4.3.5 Circuit construction diagram of PWM output prohibit function (using pin group A)

In this sample program is following two kinds of PWM output prohibition function are used.

(1) PWM output prohibit by input signal from external pin

While detecting falling edge of external signal (PWMOFF0), output to external pin TO21 to TO26 are prohibited.

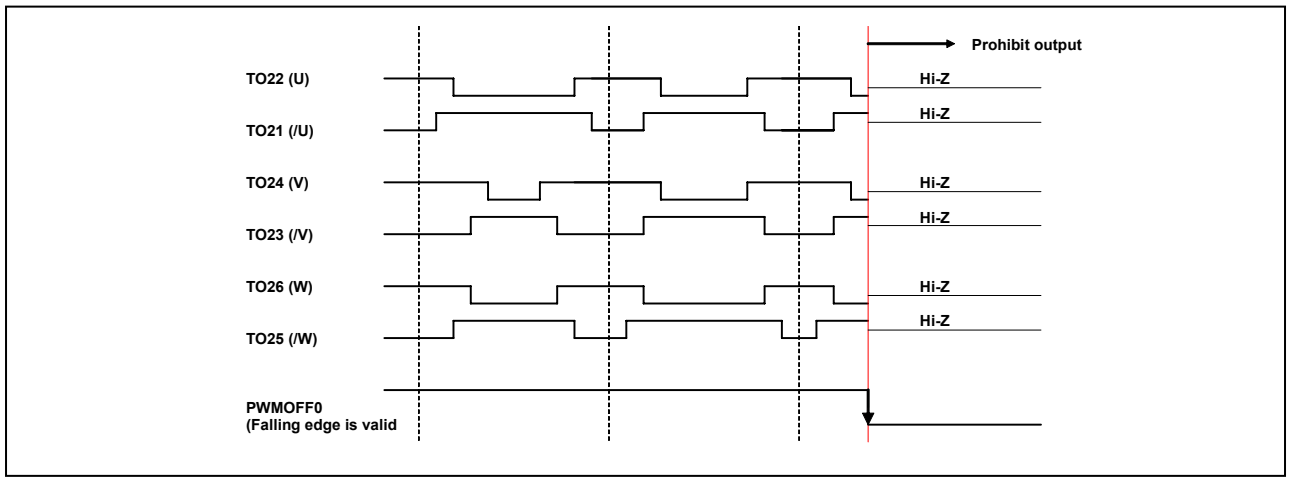


Figure 4.3.6 Operation diagram when PWM output prohibit by input signal from external pin

(2) PWM output prohibit by port pin level

While detecting simultaneous Low level in positive and negative phase of PWM waveform, PWM output to external pin TO21 to TO26 are prohibited.

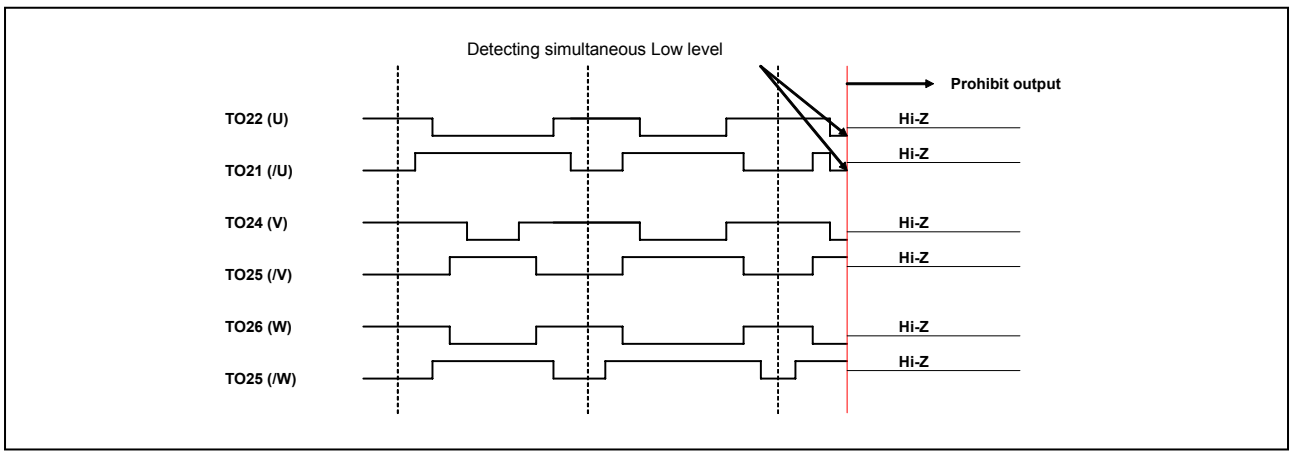


Figure 4.3.7 Operation diagram when PWM output prohibit by port pin level

4.4 Processing Procedure

Figure 4.4.1 shows the processing flow for setting the timer.

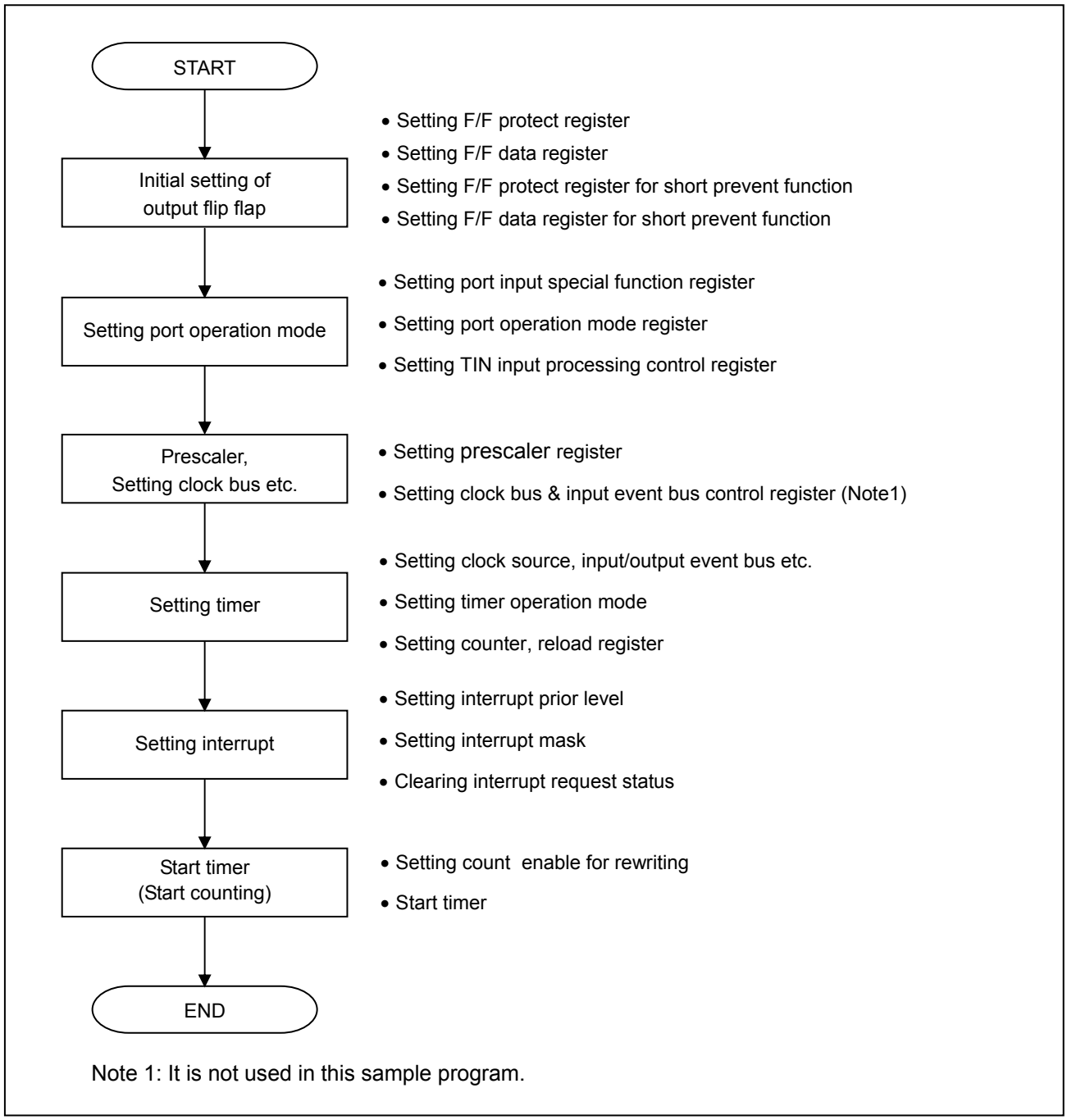

START
Initial setting of output flip flap
• Setting F/F protect register
• Setting F/F data register
• Setting F/F protect register for short prevent function
• Setting F/F data register for short prevent function
Setting port operation mode
• Setting port input special function register
• Setting port operation mode register
• Setting TIN input processing control register
Prescaler, Setting clock bus etc.
• Setting prescaler register
• Setting clock bus & input event bus control register (Note1)
Setting timer
• Setting clock source, input/output event bus etc.
• Setting timer operation mode
• Setting counter, reload register
Setting interrupt
• Setting interrupt prior level
• Setting interrupt mask
• Clearing interrupt request status
Start timer (Start counting)
• Setting count enable for rewriting
• Start timer
END
Note 1: It is not used in this sample program.

Figure 4.4.1 Timer Initialization Flow

4.5 Interpretation of the Sample Program

Note. The registers used are indicated as (register name: bit name).

4.5.1 Port initializing function (port_init())

(1) Set output flip flop control register
- Set initial output data from FF21 to FF26 (FF2128P, FF2128D)
- Set initial output data from FF21 to FF26 for short prevention function (SHFF2126P, SHFF2126D)

(2) Setting port
- Set input enable bit of port input special function control register to enable input. (PICNT: PIEN0)
- Set P11 to output mode as multipurpose port. (P11DATA, P11DIR, P11MOD)
- Set P8 to TO21 to TO26 (P8DATA, P8DIR, P8MOD, P8SMOD)
- Set P13 to TIN16 (P13DIR, P13MOD, P13SMOD)
- Set TIN16 to falling edge. (TINCR3)

4.5.2 Various initializing function (init_func())

(1) Call port initializing function

4.5.3 TOU0 initializing function (TOU0_init())

(1) Set TOU0 control register 0 as following (TOU0CR0)
- Set TOU0_0, TOU0_2, TOU0_4 to single-shot PWM output mode
- Set TOU0_1, TOU0_3, TOU0_5 to single-shot output mode
- Set TOU0_6 to single-shot output mode
- Set TOU0_7 to continuous output mode

(2) Set TOU0 control register 1 as following (TOU0CR1)
- Use prescaler 3
- Prescaler 3 supply clock is BCLK
- Set short prevention function as valid.
- Set TOU0_0, TOU0_2, TOU0_4 to single-shot PWM output mode

(3) Set operation period of TOU0_7 to counter and reload register (TOU07CTW, TOU07RLW)
(4) Set reload register as the value underflow in the point 3/4 of carrier cycle for TOU0_6 (TOU06RLW)
(5) Set prescaler divided by value
- Set “prescaler divided by value-1” to prescaler 3 (PRS3)

(6) Set TID control & prescaler enable register (TID0PRS3EN)
- Set prescaler 3 to “start”.
- Set enable factor of TOU0 to TOU0_7 underflow

(7) TOU0 PWMOFF input processing control register to falling edge (PWMOFF0CR)
(8) TOU0 PWM output 0 prohibit level control register GA (PO0LVGACR)
- Set disable output level as “L”, disable output level select as valid

(9) TOU0 PWMOFF0 function enable register to PWMOFF0 function valid (PWMOFF0EN)
(10) Set TIN interrupt control register to TIN16 enable interrupt (TINIR4, TINIR5)
(11) Set TOU0 interrupt request mask register to TOU0_6 enable interrupt request (TOU0IST, TOU0IMA)
(12) Set interrupts enable and prior level
- TOU0 output interrupt control register to interrupt level “2” (ITOU0CR)
- MJT input interrupt control register to interrupt level “0” (the highest priority) (IMJTICR2)

(13) Set shorting prevention time
- "shorting prevention time-3" is set to the reload register of the TOU0_1, TOU0_3 and TOU0_5 (TOU01RLW, TOU03RLW, TOU05RLW)

4.5.4 Duty calculation of PWM waveform and setting function of TOU0 reload register (SetDuty())

(1) Calculate “L” time positive phase side for next cycle
- Get data for 3 phase (U, V, W) from sine wave table (calculate approximately 120 degrees for phase difference).

(2) Set value for Duty 0% to FF data register for short prevention function when next Duty is 0%. (SHFF2126P, SHFF2126D)
- In order to output the PWM waveform of the smallest width, the smallest value is set to the positive phase side "L" time data

(3) Set value for Duty 100% to FF data register for short prevention function when next Duty is 100%. (SHFF2126P, SHFF2126D)
- In order to output the PWM waveform of the maximum width, the maximum value is set to the positive phase side "L" time data

(4) Set value for normal operation to FF data register for short prevention function when both next Duty is not 0% nor 100% and pervious Duty is 0% or 100%. (SHFF2126P, SHFF2126D)

(5) Set calculation result to each reload registers of TOU0 (TOU00RL1, TOU00RL0, TOU02RL1, TOU02RL0, TOU04RL1, TOU04RL0).

4.5.5 Setting function of table pointer of sine wave (SetPointer())

(1) Increment pointer to sine wave table

4.5.6 Main function (main())

(1) Initial pointer “gSinPtr” to sine wave table
(2) Initialize a coefficient of sine waveform width “gAmp” and calculate “L” time maximum value “gLlevel100Percent”.
(3) Initialize flags of Duty 0% and Duty 100% during operation
(4) Call interrupt prohibit function
(5) Call various initializing function
(6) Call TOU0 initializing function
(7) Call enable interrupt function
(8) Duty calculation of PWM waveform and call setting function of TOU0 reload register
(9) Start TOU0_7 counting (TOU0PRO, TOU0CEN)
(10) Wait interrupt infinite loop

4.5.7 TOU0 interrupt processing function (Int_TOU0_7())

(1) If interrupt request of TOU0_6 occurs, go to next processing
- Clear interrupt request of TOU0_6
- Call pointer setting function of sine wave table
- Duty calculation of PWM waveform and call setting function of TOU0 reload register

4.5.8 TIN interrupt processing function (Int_TIN1619())

(1) If interrupt request of TIN16 occurs, go to next processing
- Clear TIN16 interrupt request (TINIR4)
- All TOU0 timers are stopped. (TOU0PRO, TOU0CEN)
- Set TOU0 interrupt request mask register to TOU0_6 disable interrupt request (TOU0IST, TOU0IMA)
- TOU0 output interrupt control register to interrupt disabled (ITOU0CR)
- Set infinite loop output H’03 and H’00 alternately in P11 data register.

.

4.6 Sample Program

Sample program for Phase-compensated PWM waveform output is shown below. In this program, TOU_7 is used in continuous output mode, 20 kHz for carrier cycle.

The amplitude rate of the sine wave can be changed depending on the value of global variable “gAmp”.
When a set value of global variable “gAmp” is 100 or less, the sine wave not to include the duty 0% and the duty 100% in the PWM output is controlled.
The duty 100% is contained, and a square wave (trapezoid wave) is controlled from 100 a set value of global variable “gAmp” to the PWM output when it is large by the duty 0%.

Note that the sample programs below require the SFR definition file.
The latest SFR definition file can be downloaded from Renesas Technology website.
When using the SFR definitions file, adjust the path setting to match the operating computer environment.

4.6.1 TOU_pwm.c

```
/*""FILE COMMENT""*****************************************************
 *      M32R C Programming              Rev. 1.00
 *              < Sample Program for 32192 >
 *              < TOU0 3-phase PWM output >
 *
 *      Copyright (c) 2005 Renesas Technology Corporation
 *                         All Rights Reserved
 ***********************************************************************/

/************************************************************************/
/*              Include file                                            */
/************************************************************************/
#include               "..\inc\sfr32192_pragma.h"

/************************************************************************/
/*              Function prototype declaration                          */
/************************************************************************/
        void            port_init(void);                /* Initialize port */
        void            init_func(void);                /* Initial setup function */
        void            TOU0_init( void );              /* TOU0 initialize function */
        void            SetDuty( void );                /* Calculation of duty and register setting fu
nction */
        void            SetPointer( void );             /* Sine wave table pointer Update function */
        void            main(void);                     /* Main function */
        void            Int_TOU0_7( void );             /* TOU0 interrupt handler */
        void            Int_TIN1619( void );            /* TIN16 interrupt handler */

/************************************************************************/
/*              Definition of external reference                        */
/************************************************************************/
extern  void            DisInt( void );                 /* Interrupt disable function */
extern  void            EnInt( void );                  /* Interrupt enable function */

/************************************************************************/
/*              Define macro                                            */
/************************************************************************/
#define DEAD_TIME       120                             /* shorting prevention time (3us@40MHz(BCLK))
*/
#define CARRIER_COUNT   2000                            /* Carrier cycle (50us=20KHz@40MHz(BCLK)) */
#define AMP_MAX         40                              /* Amplitude coefficient maximum value */

#define H_MAX           (CARRIER_COUNT-(DEAD_TIME<<1)-4)/* H level max time(count clock) */
#define H_MID           ((H_MAX + H_MIN)>>1)            /* H level mid time(count clock) */
#define H_MIN           (2L)                            /* H level min time(count clock) */


#define TID0_PRS        0x07u           /* 0000 0111B   */
                                        /* |||| |||+----------- start prescaler 3 count         */
                                        /* |||| +++------------ TOU0 enable source is TOU0_7 underflow
 */
                                        /* ++++---------------- don't care                      */

#define TOU0_CR0        0x4442u         /* 0100 0100 0100 0010B */
                                        /* |||| |||| |||| ||++- TOU0_7 Continuous mode         */
                                        /* |||| |||| |||| ++--- TOU0_6 Single-shot mode        */
```

```
                                        /* |||| |||| ||++------ TOU0_5 Single-shot mode         */
                                        /* |||| |||| ++-------- TOU0_4 Single-shot pwm mode     */
                                        /* |||| ||++---------- TOU0_3 Single-shot mode         */
                                        /* |||| ++------------ TOU0_2 Single-shot pwm mode     */
                                        /* ||++-------------- TOU0_1 Single-shot mode         */
                                        /* ++---------------- TOU0_0 Single-shot pwm mode     */

#define TOU0_CR1        0x4100u         /* 0100 0001 0000 0000B */
                                        /* |||| |||| ++++ ++++- Single-shot pwm mode           */
                                        /* |||| |||+------------ enable shorting prevention function */
                                        /* ||++ +++------------- No operation                  */
                                        /* |+------------------- select BCLK                   */
                                        /* +-------------------- select prescaler3            */

/************************************************************************/
/*              Global variable                                         */
/************************************************************************/
UCHAR   gSinPtr;                                    /* pointer of sine wave table */
ULONG   gAmp;                                       /* Amplitude coefficient */
ULONG   gLlevel100Percent;                          /* L level max time(count clock) */

UCHAR   g0100FlgU;                                  /* U phase duty 0% or 100% executing flag */
UCHAR   g0100FlgV;                                  /* V phase duty 0% or 100% executing flag */
UCHAR   g0100FlgW;                                  /* W phase duty 0% or 100% executing flag */
                                                    /* 0 : normal */
                                                    /* 1 : duty 0% */
                                                    /* 2 : duty 100% */

const SSHORT gaSinTbl[ 256 ] = {                    /* sine wave table */
         0,    804,   1608,   2410,   3212,   4011,   4808,   5602,   6393,   7179,   7962,   8739,
      9512,  10278,  11039,  11793,  12539,  13279,  14010,  14732,  15446,  16151,  16846,  17530,
     18204,  18868,  19519,  20159,  20787,  21403,  22005,  22594,  23170,  23731,  24279,  24811,
     25329,  25832,  26319,  26790,  27245,  27683,  28105,  28510,  28898,  29268,  29621,  29956,
     30273,  30571,  30852,  31113,  31356,  31580,  31785,  31971,  32137,  32285,  32412,  32521,
     32609,  32678,  32728,  32757,  32767,  32757,  32728,  32678,  32609,  32521,  32412,  32285,
     32137,  31971,  31785,  31580,  31356,  31113,  30852,  30571,  30273,  29956,  29621,  29268,
     28898,  28510,  28105,  27683,  27245,  26790,  26319,  25832,  25329,  24811,  24279,  23731,
     23170,  22594,  22005,  21403,  20787,  20159,  19519,  18868,  18204,  17530,  16846,  16151,
     15446,  14732,  14010,  13279,  12539,  11793,  11039,  10278,   9512,   8739,   7962,   7179,
      6393,   5602,   4808,   4011,   3212,   2410,   1608,    804,      0,   -804,  -1608,  -2410,
     -3212,  -4011,  -4808,  -5602,  -6393,  -7179,  -7962,  -8739,  -9512, -10278, -11039, -11793,
    -12539, -13279, -14010, -14732, -15446, -16151, -16846, -17530, -18204, -18868, -19519, -20159,
    -20787, -21403, -22005, -22594, -23170, -23731, -24279, -24811, -25329, -25832, -26319, -26790,
    -27245, -27683, -28105, -28510, -28898, -29268, -29621, -29956, -30273, -30571, -30852, -31113,
    -31356, -31580, -31785, -31971, -32137, -32285, -32412, -32521, -32609, -32678, -32728, -32757,
    -32767, -32757, -32728, -32678, -32609, -32521, -32412, -32285, -32137, -31971, -31785, -31580,
    -31356, -31113, -30852, -30571, -30273, -29956, -29621, -29268, -28898, -28510, -28105, -27683,
    -27245, -26790, -26319, -25832, -25329, -24811, -24279, -23731, -23170, -22594, -22005, -21403,
    -20787, -20159, -19519, -18868, -18204, -17530, -16846, -16151, -15446, -14732, -14010, -13279,
    -12539, -11793, -11039, -10278,  -9512,  -8739,  -7962,  -7179,  -6393,  -5602,  -4808,  -4011,
    -3212,  -2410,  -1608,   -804
 };

/*""FUNC COMMENT""*************************************************************
 * Function name: port_init()
 *-----------------------------------------------------------------------
 * Description  : Initialize port
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : -
 *""FUNC COMMENT END""*********************************************************/
void port_init(void)
{
        /* Setting F/F */
        FF2128P = 0x00;
        FF2128D = 0x54;                             /* 01010100B */
        SHFF2126P = 0x00;
        SHFF2126D = 0xA8;                           /* 10101000B */

        PICNT = PIEN0;                              /* Enable port input */

        /*** Error Output (LED) ***/
        P11DATA = 0x00;                             /* Output data (must be set prior to mode) */
        P11DIR = 0xff;                              /* P110-P117 : Output mode */
```

```
130         P11MOD = 0x00;                                  /* P110-P117 : Input/output port */
131
132         /*** Setting TO21-TO26 ***/
133         P8DATA = 0x00;                                  /* Output data (must be set prior to mode) */
134         P8DIR  = 0xff;                                  /* P82-P87 : Output mode */
135         P8MOD  = 0x00;                                  /* P82-P87 : clear */
136         P8SMOD = 0x3f;                                  /* P82-P87 : TOn select */
137         P8MOD  = 0x3f;                                  /* P82-P87 : TO26 - TO21 */
138
139         /*** Setting TIN16 ***/
140         P13DIR  = 0x00;                                 /* P130-P137 : Input mode */
141         P13MOD  = 0x00;
142         P13SMOD = 0x80;                                 /* Select P130 for TIN16 */
143         P13MOD |= 0x80u;                                /* Select P130 for TIN16 */
144         TINCR3  = 0x0200;                               /* TIN16 : falling edge */
145 }
146
147 /*""FUNC COMMENT""*************************************************
148  * Function name: init_func()
149  *-----------------------------------------------------------------
150  * Description  : Call various initialization functions
151  *-----------------------------------------------------------------
152  * Argument     : -
153  *-----------------------------------------------------------------
154  * Returns      : -
155  *-----------------------------------------------------------------
156  * Notes        : -
157  *""FUNC COMMENT END""*************************************************/
158 void init_func(void)
159 {
160         port_init();
161 }
162
163 /*""FUNC COMMENT""*************************************************
164  * Function name: TOU0_init()
165  *-----------------------------------------------------------------
166  * Description  : -
167  *-----------------------------------------------------------------
168  * Argument     : -
169  *-----------------------------------------------------------------
170  * Returns      : -
171  *-----------------------------------------------------------------
172  * Notes        : The prescaler, clock bus, etc. are set separately from the above
173  *              : Must be executed while interrupts are disabled
174  *""FUNC COMMENT END""*************************************************/
175 void    TOU0_init( void )
176 {
177         TOU0CR0 = TOU0_CR0;                             /* Set TOU0_n operation mode */
178         TOU0CR1 = TOU0_CR1;                             /* Set supplied clock select BCLK */
179
180         TOU07CTW = CARRIER_COUNT - 1ul;
181         TOU07RLW = CARRIER_COUNT - 1ul;                 /* TOU0_7 reload register */
182
183         TOU06RLW = (CARRIER_COUNT * 3ul / 4ul) - 80ul;  /* TOU0_6 use for data setting */
184                                                         /* 80 : Interrupt processing time */
185
186         PRS3 = ( 1 - 1);                                /* Set prescaler3(25ns@40MHz(BCLK)) */
187         TID0PRS3EN = TID0_PRS;
188
189         /* Setting PWM OFF */
190         PWMOFF0CR = PWMOFF0SP;                          /* PWMOFF0 writing control */
191         PWMOFF0CR = 0x02;                               /* PWMOFF0 falling edge */
192         PO0LVGACR = 0x00;                               /* "L"level enable */
193         PWMOFF0EN = 0x40;                               /* P87 - P82 PWM OFF enable */
194
195         /* Setting interrupt */
196         TINIR4 = 0x00;                                  /* clear interrupt request */
197         TINIR5 = (~TINIM16) & 0xffu;                    /* enable TIN16 */
198         TOU0IST = 0x00;                                 /* clear interrupt request */
199         TOU0IMA = (~TOU0IM6) & 0xffu;                   /* enable TOU0_6 */
200         ITOU0CR = 0x02;                                 /* TOU0_n interrupt enable */
201         IMJTICR2 = 0x00;                                /* TIN16 interrupt enable */
202
203         /* Setting shorting prevention time */
204         TOU01RLW = DEAD_TIME - 3ul;
205         TOU03RLW = DEAD_TIME - 3ul;
206         TOU05RLW = DEAD_TIME - 3ul;
```

```
}

/*""FUNC COMMENT""*************************************************************
 * Function name: void SetDuty( void )
 *-------------------------------------------------------------------------------
 * Description : Next cycle duty calculation and setting of reload registers
 *-------------------------------------------------------------------------------
 * Argument    : none
 *-------------------------------------------------------------------------------
 * Returns     : none
 *-------------------------------------------------------------------------------
 * Notes       : The change from 0 to 100% and the change from 100 to 0% are the prohibitions.
 *""FUNC COMMENT END""*********************************************************/
void SetDuty( void )
{
        UCHAR   SinPtrU, SinPtrV;
        USHORT  DutyU_RL0, DutyU_RL1;
        USHORT  DutyV_RL0, DutyV_RL1;
        USHORT  DutyW_RL0, DutyW_RL1;
        SSHORT  OrdVu, OrdVv, OrdVw;
        SLONG   HtimeU, HtimeV, HtimeW;

        /** Get sine wave table pointer **/
        SinPtrU = gSinPtr;
        SinPtrV = gSinPtr + 86;
        /** U-phase increase and decrease amount calculation of "H" time **/
        OrdVu = (gLlevel100Percent * gaSinTbl[ SinPtrU ]) >> 16;
        HtimeU = OrdVu + H_MID;
        /** V-phase increase and decrease amount calculation of "H" time **/
        OrdVv = (gLlevel100Percent * gaSinTbl[ SinPtrV ]) >> 16;
        HtimeV = OrdVv + H_MID;
        /** W-phase increase and decrease amount calculation of "H" time **/
        OrdVw = - OrdVu - OrdVv;
        HtimeW = OrdVw + H_MID;

        /** U-phase duty judgment and setting at the next cycle **/
        if( HtimeU < H_MIN ){                           /* Duty 0% */
                HtimeU = H_MIN;
                if( g0100FlgU == 0 ){                   /* /U="H", U="L" fix */
                        SHFF2126P = 0x3F;
                        g0100FlgU = 1;                  /* Duty 0% memory */
                        SHFF2126D = 0x40u;              /* '01xx xxxx' write */
                }
        }
        else if( H_MAX < HtimeU ){                      /* Duty 100% */
                HtimeU = H_MAX;
                if( g0100FlgU == 0 ){                   /* /U="L", U="H" fix */
                        SHFF2126P = 0x3F;
                        g0100FlgU = 2;                  /* Duty 100% memory */
                        SHFF2126D = 0x80u;              /* '10xx xxxx' write */
                }
        }
        else{                                           /* normal */
                if( g0100FlgU == 1 ){                   /* It was duty 0% last time */
                        SHFF2126P = 0x3F;
                        g0100FlgU = 0;
                        SHFF2126D = 0x80u;              /* '01xx xxxx' write */
                }
                else if( g0100FlgU == 2 ){              /* It was duty 100% last time */
                        SHFF2126P = 0x3F;
                        g0100FlgU = 0;
                        SHFF2126D = 0x40u;              /* '10xx xxxx' write */
                }
        }
        DutyU_RL1 = HtimeU + DEAD_TIME - 1l;
        DutyU_RL0 = ((CARRIER_COUNT - HtimeU) >> 1) - DEAD_TIME - 2l;

        /** V-phase duty judgment and setting at the next cycle **/
        if( HtimeV < H_MIN ){                           /* Duty 0% */
                HtimeV = H_MIN;
                if( g0100FlgV == 0 ){                   /* /V="H", V="L" fix */
                        SHFF2126P = 0xCF;
                        g0100FlgV = 1;                  /* Duty 0% memory */
                        SHFF2126D = 0x10;               /* 'xx10 xxxx' write */
                }
        }
        else if( H_MAX < HtimeV ){                      /* Duty 100% */
```

```
284                 HtimeV = H_MAX;
285                 if( g0100FlgV == 0 ){                   /* /V="L", V="H" fix */
286                         SHFF2126P = 0xCF;
287                         g0100FlgV = 2;                  /* Duty 100% memory */
288                         SHFF2126D = 0x20;               /* 'xx01 xxxx' write */
289                 }
290         }
291         else{                                           /* normal */
292                 if( g0100FlgV == 1 ){                   /* It was duty 0% last time */
293                         SHFF2126P = 0xCF;
294                         g0100FlgV = 0;
295                         SHFF2126D = 0x20;               /* 'xx01 xxxx' write */
296                 }
297                 else if( g0100FlgV == 2 ){              /* It was duty 100% last time */
298                         SHFF2126P = 0xCF;
299                         g0100FlgV = 0;
300                         SHFF2126D = 0x10;               /* 'xx10 xxxx' write */
301                 }
302         }
303         DutyV_RL1 = HtimeV + DEAD_TIME - 1l;
304         DutyV_RL0 = ((CARRIER_COUNT - HtimeV) >> 1) - DEAD_TIME - 2l;
305
306         /** W-phase duty judgment and setting at the next cycle **/
307         if( HtimeW < H_MIN ){                           /* Duty 0% */
308                 HtimeW = H_MIN;
309                 if( g0100FlgW == 0 ){                   /* /W="H", W="L" fix */
310                         SHFF2126P = 0xF3;
311                         g0100FlgW = 1;                  /* Duty 0% memory */
312                         SHFF2126D = 0x04u;              /* 'xxxx 10xx' write */
313                 }
314         }
315         else if( H_MAX < HtimeW ){                      /* Duty 100% */
316                 HtimeW = H_MAX;
317                 if( g0100FlgW == 0 ){                   /* /W="L", W="H" fix */
318                         SHFF2126P = 0xF3;
319                         g0100FlgW = 2;                  /* Duty 100% memory */
320                         SHFF2126D = 0x08u;              /* 'xxxx 01xx' write */
321                 }
322         }
323         else{                                           /* normal */
324                 if( g0100FlgW == 1 ){                   /* It was duty 0% last time */
325                         SHFF2126P = 0xF3;
326                         g0100FlgW = 0;
327                         SHFF2126D = 0x08u;              /* 'xxxx 01xx' write */
328                 }
329                 else if( g0100FlgW == 2 ){              /* It was duty 100% last time */
330                         SHFF2126P = 0xF3;
331                         g0100FlgW = 0;
332                         SHFF2126D = 0x04u;              /* 'xxxx 10xx' write */
333                 }
334         }
335         DutyW_RL1 = HtimeW + DEAD_TIME - 1l;
336         DutyW_RL0 = ((CARRIER_COUNT - HtimeW) >> 1) - DEAD_TIME - 2l;
337
338         /** Data setting **/
339         TOU00RL1 = DutyU_RL1;
340         TOU00RL0 = DutyU_RL0;
341         TOU02RL1 = DutyV_RL1;
342         TOU02RL0 = DutyV_RL0;
343         TOU04RL1 = DutyW_RL1;
344         TOU04RL0 = DutyW_RL0;
345 }
346
347 /*""FUNC COMMENT""*****************************************************************
348  * Function name: void SetPointer( void )
349  *-------------------------------------------------------------------------------
350  * Description : setting pointer of sine wave table
351  *-------------------------------------------------------------------------------
352  * Argument    : none
353  *-------------------------------------------------------------------------------
354  * Returns     : none
355  *-------------------------------------------------------------------------------
356  * Notes       : Only the increment is done for the simplification.
357  *""FUNC COMMENT END""*************************************************************/
358 void SetPointer( void )
359 {
360         gSinPtr++;
```

```
361  }
362
363  /*""FUNC COMMENT""************************************************
364   * Function name: main()
365   *-----------------------------------------------------------------
366   * Description  : -
367   *-----------------------------------------------------------------
368   * Argument     : -
369   *-----------------------------------------------------------------
370   * Returns      : -
371   *-----------------------------------------------------------------
372   * Notes        : -
373   *""FUNC COMMENT END""*************************************************/
374  void main(void)
375  {
376          gSinPtr = 0;
377          gAmp = 40ul;                                    /* Amplitude coefficient */
378
379          /* When the gAmp data is changed, it is necessary to calculate the "gLlevel100Percent" again *
/
380          if( gAmp > AMP_MAX ){
381                  gAmp = AMP_MAX;
382          }
383          gLlevel100Percent = gAmp * H_MAX / 100;         /* L level max value(count clock) */
384
385          g0100FlgU = 0;                                  /* duty 0% or 100% executing flags clear */
386          g0100FlgV = 0;
387          g0100FlgW = 0;
388
389          DisInt();                                       /* Disable interrupt */
390
391          init_func();
392
393          TOU0_init();                                    /* Initialize TOU0 */
394
395          EnInt();                                        /* Enable interrupt */
396
397          SetDuty();
398
399          TOU0PRO = 0xFE;                                 /* TOU0_7 write enable */
400          TOU0CEN = 0x01;                                 /* TOU0_7 start */
401
402          while( 1 ){
403                  ;
404          }
405  }
406
407  /*""FUNC COMMENT""************************************************************
408   * Function name: void Int_TOU0_7( void )
409   *-------------------------------------------------------------------------------
410   * Description : TOU0 counter under flow interrupt
411   *-------------------------------------------------------------------------------
412   * Argument    : none
413   *-------------------------------------------------------------------------------
414   * Returns     : none
415   *-------------------------------------------------------------------------------
416   * Notes       : -
417   *""FUNC COMMENT END""*********************************************************/
418  void Int_TOU0_7( void )
419  {
420          /* TOU0_6 interrupt */
421          if( (TOU0IST & TOU0IS6) != 0u ){
422                  TOU0IST = ~TOU0IS6 & 0xffu;             /* clear TOU0_6 interrupt request */
423                  SetPointer();
424                  SetDuty();
425          }
426  }
427
428  /*""FUNC COMMENT""************************************************************
429   * Function name: void Int_TIN1619( void )
430   *-------------------------------------------------------------------------------
431   * Description : TIN16 - TIN19 interrupt request
432   *-------------------------------------------------------------------------------
433   * Argument    : none
434   *-------------------------------------------------------------------------------
435   * Returns     : none
436   *-------------------------------------------------------------------------------
```

```
437   * Notes       : FO signal input proc
438   *""FUNC COMMENT END""*****************************************************/
439  void Int_TIN1619( void )
440  {
441          if ( (TINIR4 & TINIS16) != 0u ){
442                  TINIR4 = ~TINIS16 & 0xffu;              /* clear TIN16 interrupt request */
443
444                  TOU0PRO = 0x00;                         /* TOU0_0-7 write enable */
445                  TOU0CEN = 0x00;                         /* TOU0_0-7 stop */
446
447                  TOU0IST = 0x00;                         /* clear interrupt request */
448                  TOU0IMA = 0xff;                         /* disable TOU0_6 */
449                  ITOU0CR = 0x07;                         /* TOU0_n interrupt disable */
450
451                  /** Error output for LED **/
452                  while(1){
453                          SINT32 i;
454                          for(i=0;i<0x100000;i++){
455                                  ;                       /* wait */
456                          }
457                          P11DATA^=0x03u;
458                  }
459          }
460  }
```

4.6.2 startup.ms (an extract)

(Omitted)

```
   72  ;*************************************
   73  ; ICU Vector Table
   74  ;*************************************
   75  ;
   76          .SECTION        ICUVECT, DATA, ALIGN=4
   77  ;
   78          .IMPORT         $Int_TIN1619
   79          .IMPORT         $Int_TOU0_7
   80  ;
   81  vectbl:
   82          .DATA.W         EIT_reset           ; H'0000 0094   MJT Input Interrupt 4:TIN3-TIN6
   83          .DATA.W         EIT_reset           ; H'0000 0098   MJT Input Interrupt 3:TIN20-TIN27
   84          .DATA.W         $Int_TIN1619        ; H'0000 009C   MJT Input Interrupt 2:TIN16-TIN19
   85          .DATA.W         EIT_reset           ; H'0000 00A0   MJT Input Interrupt 1:TIN0
   86          .DATA.W         EIT_reset           ; H'0000 00A4   MJT Input Interrupt 0:TIN7-TIN10
   87          .DATA.W         EIT_reset           ; H'0000 00A8   MJT Output Interrupt 7:TMS0,TMS1
   88          .DATA.W         EIT_reset           ; H'0000 00AC   MJT Output Interrupt 6:TOP8,TOP9
   89          .DATA.W         EIT_reset           ; H'0000 00B0   MJT Output Interrupt 5:TOP10
   90          .DATA.W         EIT_reset           ; H'0000 00B4   MJT Output Interrupt 4:TIO4-TIO7
   91          .DATA.W         EIT_reset           ; H'0000 00B8   MJT Output Interrupt 3:TIO8,TIO9
   92          .DATA.W         EIT_reset           ; H'0000 00BC   MJT Output Interrupt 2:TOP0-TOP5
   93          .DATA.W         EIT_reset           ; H'0000 00C0   MJT Output Interrupt 1:TOP6,TOP7
   94          .DATA.W         EIT_reset           ; H'0000 00C4   MJT Output Interrupt 0:TIO0-TIO3
   95          .DATA.W         EIT_reset           ; H'0000 00C8   DMAC0-4 Interrupt:DMA0-DMA4
   96          .DATA.W         EIT_reset           ; H'0000 00CC   SIO1 Receive Interrupt
   97          .DATA.W         EIT_reset           ; H'0000 00D0   SIO1 Transmit Interrupt
   98          .DATA.W         EIT_reset           ; H'0000 00D4   SIO0 Receive Interrupt
   99          .DATA.W         EIT_reset           ; H'0000 00D8   SIO0 Transmit Interrupt
  100          .DATA.W         EIT_reset           ; H'0000 00DC   A/D0 Conversion Interrupt
  101          .DATA.W         EIT_reset           ; H'0000 00E0   TID0 Output Interrupt
  102          .DATA.W         $Int_TOU0_7         ; H'0000 00E4   TOD0 Output Interrupt
  103          .DATA.W         EIT_reset           ; H'0000 00E8   DMAC5-9 Interrupt:DMA5-DMA9
  104          .DATA.W         EIT_reset           ; H'0000 00EC   SIO2,3 Transmit/Receive Interrupt
  105          .DATA.W         EIT_reset           ; H'0000 00F0   RTD Interrupt
  106          .DATA.W         EIT_reset           ; H'0000 00F4   TID1 Output Interrupt
  107          .DATA.W         EIT_reset           ; H'0000 00F8   TOU1 Output Interrupt:TOU1_0-
TOU1_7
  108          .DATA.W         EIT_reset           ; H'0000 00FC   SIO4,5 Transmit/Receive Interrupt
  109          .DATA.W         EIT_reset           ; H'0000 0100   Reserved
  110          .DATA.W         EIT_reset           ; H'0000 0104   Reserved
  111          .DATA.W         EIT_reset           ; H'0000 0108   TML1 Input Interrupt:TIN30-TIN33
  112          .DATA.W         EIT_reset           ; H'0000 010C   CAN0 Transmit/Receive & Error
Interrupt
  113          .DATA.W         EIT_reset           ; H'0000 0110   CAN1 Transmit/Receive & Error
Interrupt
  114          .DATA.W         EIT_reset           ; H'0000 0114   DRI Transfer Interrupt
  115          .DATA.W         EIT_reset           ; H'0000 0118   DRI Counter Interrupt:DEC0-DEC4
  116          .DATA.W         EIT_reset           ; H'0000 011C   DRI Event Detection
Interrupt:DIN0-DIN5
  117          .DATA.W         EIT_reset           ; H'0000 0120   CAN0 Transmit/Receive Completion
Interrupt
  118          .DATA.W         EIT_reset           ; H'0000 0124   CAN0 Single-Shot Interrupt
  119          .DATA.W         EIT_reset           ; H'0000 0128   CAN0 Error Interrupt
  120          .DATA.W         EIT_reset           ; H'0000 012C   CAN1 Transmit/Receive Completion
Interrupt
  121          .DATA.W         EIT_reset           ; H'0000 0130   CAN1 Single-Shot Interrupt
  122          .DATA.W         EIT_reset           ; H'0000 0134   CAN1 Error Interrupt
  123          .DATA.W         EIT_reset           ; H'0000 0138   RAM Write Monitor Interrupt
  124  ;
```

(Omitted)

5. Reference Documents

- 32192/32196 Group Hardware Manual (Rev.1.01)
- 32195 Group Datasheet (Rev.1.00)
- M3T-CC32R Version 5.00 User's Manual (Compiler)
- M3T-AS32R Version 5.00 User's Manual (Assembler)
- M32R-FPU Software Manual (Rev.1.01)

(Please get the latest one from Renesas Technology Corp. website.)

6. Homepage and Support Center

- Renesas Technology Corp. Website:
  http://www.renesas.com/

- Inquires for all Renesas products and technical inquiries for the M32R Family products:
  Customer Support Center: csc@renesas.com

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Feb.22.06	—	First edition issued.

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.